Web Scraping & Sentiment analysis

By: Priya Yadav

Introduction

- The project integrates web scraping with sentiment analysis to achieve the following objectives:
- Extract textual data from specified web pages.
- Analyze the sentiment of the extracted content to determine whether it is positive, negative, or neutral.
- Identify key insights and trends from the data, such as common themes or recurring sentiments.

Tools and Libraries Used Technologies:

- **csv:** For handling CSV files to store scraped data.
- **Flask, render_template, request:** Flask modules for web application functionality.
- **time:** Used for delays in web scraping.
- **selenium.webdriver:** Optional for browser automation.
- **transformers:** Hugging Face library for NLP tasks.
- **requests:** For making HTTP requests.
- **string, nltk:** Text preprocessing utilities.
- **torch:** For tensor operations.
- **TextBlob:** For sentiment analysis.

Preprocess_text

- **Function Purpose:**
  - Cleans and preprocesses text data.
  - Converts text to lowercase, removes punctuation and stopwords, and performs lemmatization.

```python
def preprocess_text(text):
    """
    Preprocesses text by removing punctuation, stopwords, and lemmatizing.
    """

    text = text.lower()  # Convert to lowercase
    text = ''.join([c for c in text if c not in string.punctuation])  # Remove punctuatio
    tokens = word_tokenize(text)
    stop_words = stopwords.words('english')
    filtered_words = [w for w in tokens if w not in stop_words]
    lemmatizer = WordNetLemmatizer()
    lemmatized_words = [lemmatizer.lemmatize(word) for word in filtered_words]
```

Data Handling - CSV Storage

- **– Data Storage:**
- Saves scraped data (URL, sentiment analysis results, hashtags, content) to a CSV file (`scrapeddata.csv`).
- Uses `csv.writer` for efficient data storage and retrieval.

main.py | scrapeddata.csv | ss.py | index.html | result.html

*.csv files are supported by PyCharm Professional Try PyCharm Profes

```
url,sentiment,polarity,subjectivity,hashtags ,insights
https://www.amazon.in/s/?_encoding=UTF8&rh=i%3Aindustrial%2Cn%3A12109315
https://timesofindia.indiatimes.com/,positive,0.350694444,0.642708333,[]
https://www.amazon.in/?&tag=googhydrabk1-21&ref=pd_sl_7hz2t19t5c_e&adgrp
https://economictimes.indiatimes.com/markets/stocks?from=mdr,negative,0.
https://www.instagram.com/p/C6Vy1v5Lmnt/?igsh=MTc4MmM1YmI2Ng==,positive,
https://youtu.be/HjsQsjX38i0?si=8HQrR7iN53sMAtAR,positive,0,0,[],
https://www.youtube.com/watch?v=RBSUwFGa6Fk,positive,0,0,[],
"https://www.flipkart.com/all/~cs-af3fbf7e18ed75a9cbc2a9c0c37bd2e2/pr?si
```

Data Processing and Categorization:

- **Objective**: To process and categorize the scraped data for further analysis.

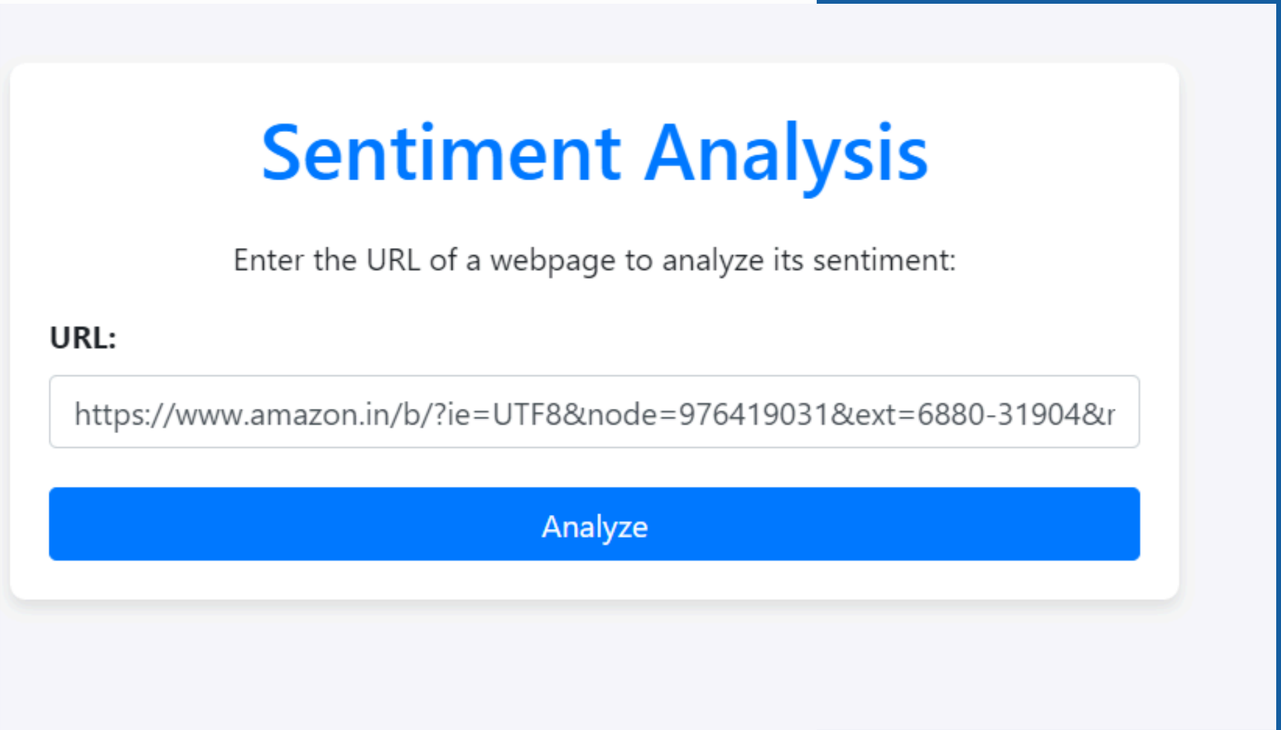
- **Categorization:** URLs are categorized into predefined categories such as Electronics, Fashion, Books, Home & Garden, Health & Fitness, E-commerce, News, Education, and Storage based on patterns in the URLs.

Flask Application - Index and Result Rendering

- Index Page:

- Displays a form for entering a URL and an option for using Selenium.

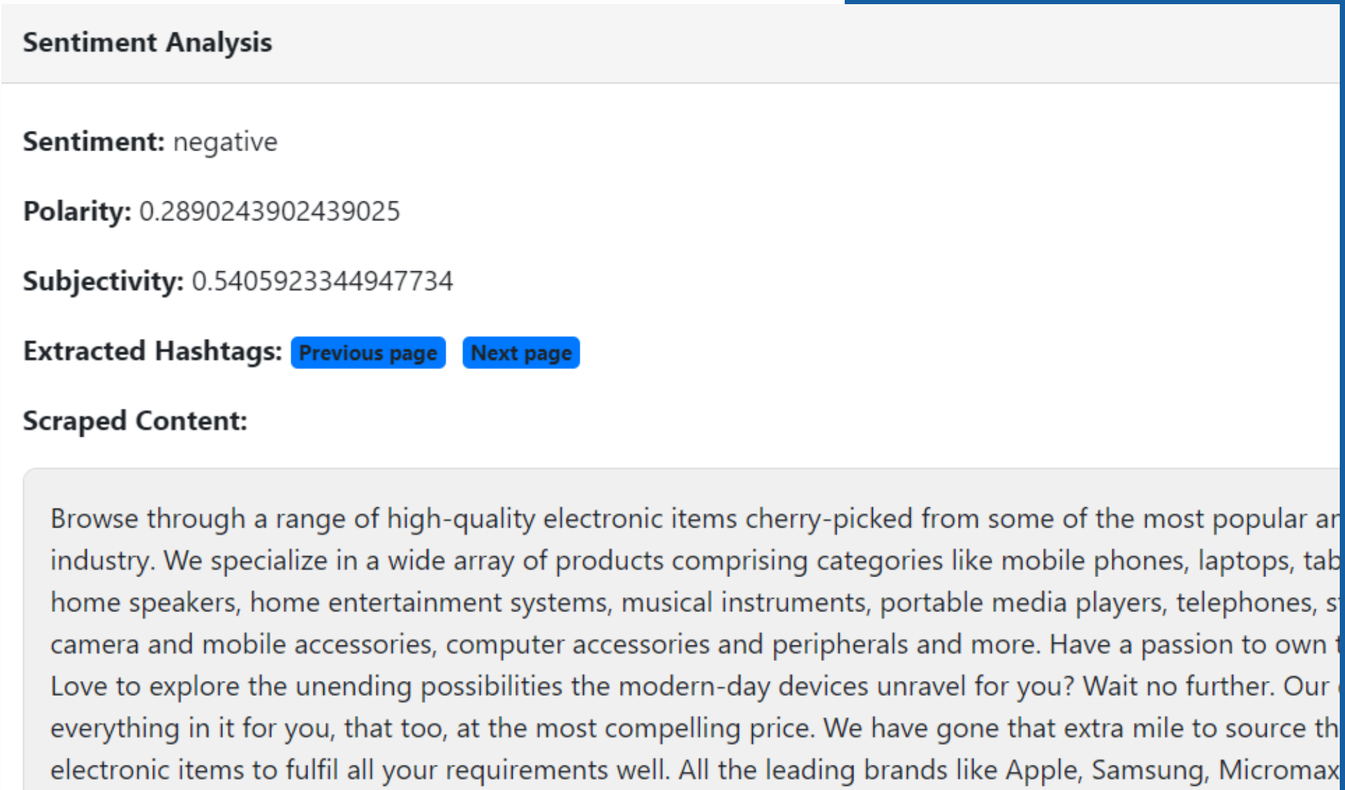
- Handles form submission to initiate web scraping and sentiment analysis.



Flask Application - Index and Result Rendering

- Result Page:

- Renders sentiment analysis results (`render_template('result.html', ...)`) including URL, sentiment, polarity, subjectivity, hashtags, and scraped content.

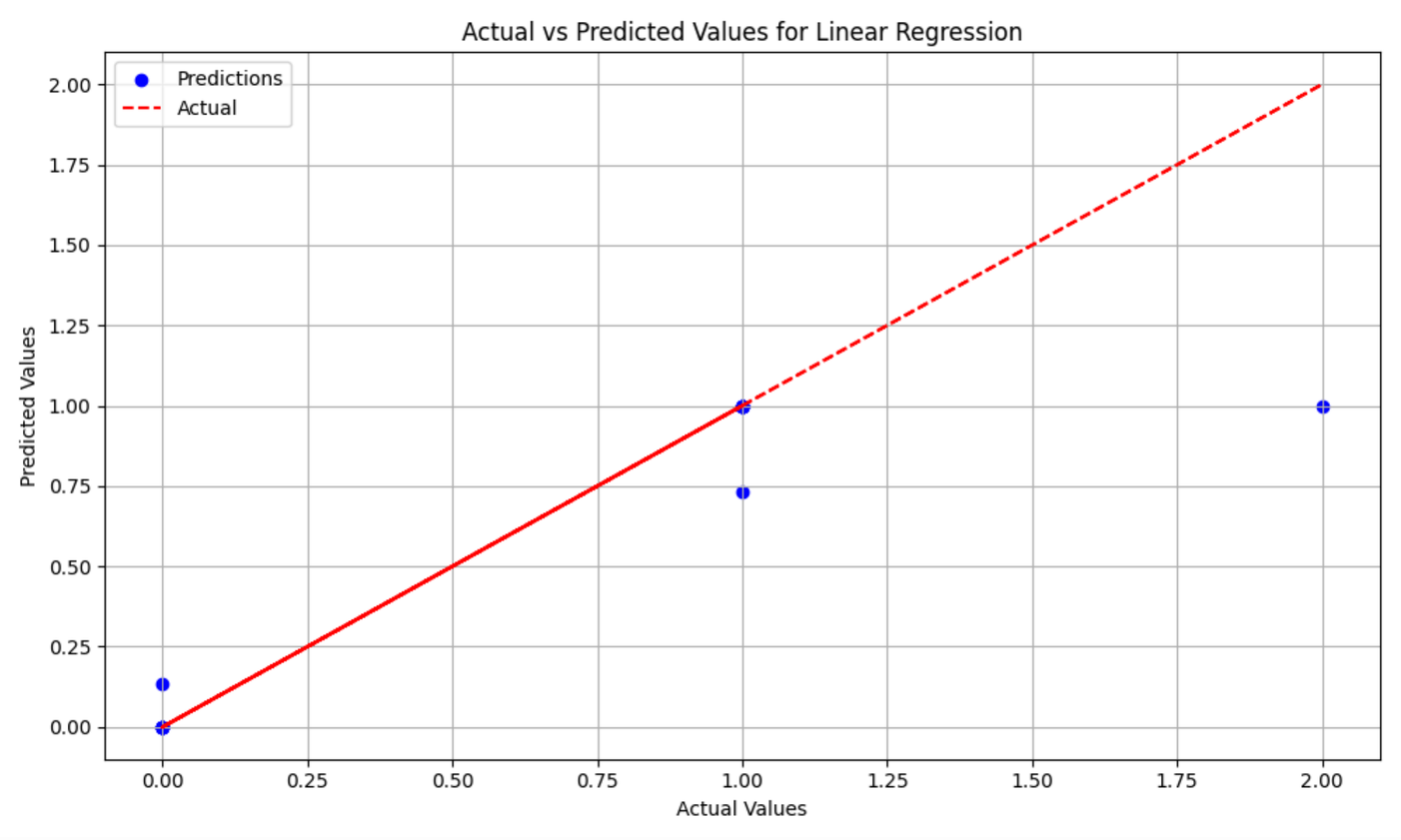


Best Model Selection

Best Model: Linear Regression

Model Metrics:

- Linear Regression - MSE: 0.06054566154300783, R2: 0.8183630153709766
- Decision Tree - MSE: 0.1111111111111111, R2: 0.6666666666666667
- Random Forest - MSE: 0.07136111111111111, R2: 0.7859166666666667
- Support Vector Machine - MSE: 0.10307820085716768, R2: 0.6907653974284971
- K-Nearest Neighbors - MSE: 0.07777777777777778, R2: 0.7666666666666667



Challenges and Solutions

- Challenges Faced:

- Handling dynamic content with Selenium.
- Dealing with anti-scraping measures.
- Ensuring accurate sentiment analysis results.

- Solutions:

- Implementing delays (`time.sleep()`) and error handling.
- Fine-tuning NLP models for better accuracy.
- Using robust libraries like BeautifulSoup and Transformers.

Future Enhancements

- Future Work:

- Improving scalability for large-scale scraping.

- Enhancing sentiment analysis with domain-specific models.

- Implementing user authentication and data privacy features.

THANK YOU!

Resource Page